

Universidade Federal do Piauí – UFPI
Centro de Ciência da Natureza – CCN
Departamento de Computação – DC
Bacharelado em Ciência da Computação

Java Collections

- Relatório para a matéria de Estrutura de Dados II sobre a aplicação criada que exemplifica e testa a classe Collection do Java, com as operações de inserção e busca.

Paulo Eduardo Ramos de Araújo
Amós Lima Magalhães
Teresina – Piauí
24/06/2019

Desde as primeiras versões, Java dispõe das estruturas de arrays e as classes Vector e Hashtable. No entanto, além da dificuldade em implementar estruturas de dados utilizando arrays, os desenvolvedores sentiam falta de classes que implementassem estruturas como listas ligadas e tabelas de espalhamento (hash). Para atender a essas necessidades, a partir de Java 1.2, foi criado um conjunto de interfaces e classes denominado Collections Framework, que faz parte do pacote java.util.

Interfaces

- Collection – está no topo da hierarquia. Não existe implementação direta dessa interface, mas ela define as operações básicas para as coleções, como adicionar, remover, esvaziar, etc.;
- Set – interface que define uma coleção que não permite elementos duplicados. A interface SortedSet, que estende Set, possibilita a classificação natural dos elementos, tal como a ordem alfabética;
- List – define uma coleção ordenada, podendo conter elementos duplicados. Em geral, o usuário tem controle total sobre a posição onde cada elemento é inserido e pode recuperá-los através de seus índices. Prefira esta interface quando precisar de acesso aleatório, através do índice do elemento;
- Queue – um tipo de coleção para manter uma lista de prioridades, onde a ordem dos seus elementos, definida pela implementação de Comparable ou Comparator, determina essa prioridade. Com a interface fila pode-se criar filas e pilhas;
- Map – mapeia chaves para valores. Cada elemento tem na verdade dois objetos: uma chave e um valor. Valores podem ser duplicados, mas chaves não. SortedMap é uma interface que estende Map, e permite classificação ascendente das chaves. Uma aplicação dessa interface é a classe Properties, que é usada para persistir propriedades/configurações de um sistema, por exemplo.

Implementações

Implementações					
Interfaces	Tabela de Espalhamento	Array Redimensionável	Árvore	Lista Ligada	Tabela de Espalhamento + Lista Ligada
Set	HashSet		TreeSet		LinkedHashSet
List		ArrayList		LinkedList	
Queue					
Map	HashMap		TreeMap		LinkedHashMap

- LinkedList – implementa uma lista ligada, ou seja, cada nó contém o dado e uma referência para o próximo nó. Ao contrário do ArrayList, a busca é linear e inserções e exclusões são rápidas. Portanto, prefira LinkedList quando a aplicação exigir grande quantidade de inserções e exclusões. Um pequeno sistema para gerenciar suas compras mensais de supermercado pode ser uma boa aplicação, dada a necessidade de constantes inclusões e exclusões de produtos.

- **ArrayList** – é como um array cujo tamanho pode crescer. A busca de um elemento é rápida, mas inserções e exclusões não são. Podemos afirmar que as inserções e exclusões são lineares, o tempo cresce com o aumento do tamanho da estrutura. Esta implementação é preferível quando se deseja acesso mais rápido aos elementos. Por exemplo, se você quiser criar um catálogo com os livros de sua biblioteca pessoal e cada obra inserida receber um número sequencial (que será usado para acesso) a partir de zero.
- **HashSet** – o acesso aos dados é mais rápido que em um **TreeSet**, mas nada garante que os dados estarão ordenados. Escolha este conjunto quando a solução exigir elementos únicos e a ordem não for importante. Poderíamos usar esta implementação para criar um catálogo pessoal das canções da nossa discografia;
- **LinkedHashSet** – é derivada de **HashSet**, mas mantém uma lista duplamente ligada através de seus itens. Seus elementos são iterados na ordem em que foram inseridos. Opcionalmente é possível criar um **LinkedHashSet** que seja percorrido na ordem em que os elementos foram acessados na última iteração. Poderíamos usar esta implementação para registrar a chegada dos corredores de uma maratona;
- **TreeSet** – os dados são classificados, mas o acesso é mais lento que em um **HashSet**. Se a necessidade for um conjunto com elementos não duplicados e acesso em ordem natural, prefira o **TreeSet**. É recomendado utilizar esta coleção para as mesmas aplicações de **HashSet**, com a vantagem dos objetos estarem em ordem natural;
- **HashMap** – baseada em tabela de espalhamento, permite chaves e valores null. Não existe garantia que os dados ficarão ordenados. Escolha esta implementação se a ordenação não for importante e desejar uma estrutura onde seja necessário um ID (identificador).
- **LinkedHashMap** – mantém uma lista duplamente ligada através de seus itens. A ordem de iteração é a ordem em que as chaves são inseridas no mapa. Se for necessário um mapa onde os elementos são iterados na ordem em que foram inseridos, use esta implementação. O registro dos corredores de uma maratona, onde a chave seria o número que cada um recebe no ato da inscrição, é um exemplo de aplicação desta coleção.
- **TreeMap** – implementa a interface **SortedMap**. Há garantia que o mapa estará classificado em ordem ascendente das chaves. Mas existe a opção de especificar uma ordem diferente. Use esta implementação para um mapa ordenado. Aplicação semelhante a **HashMap**, com a diferença que **TreeMap** perde no quesito desempenho;
- **B-Tree** – Árvores B são uma generalização das árvores binária de busca, pois cada nó de uma árvore binária armazena uma única chave de busca, enquanto as árvores B armazenam um número maior do que um de chaves de busca em cada nó, ou no termo mais usual para essa árvore, em cada página. Sua busca e inserção é relativamente rápida e seguem a complexidade $O(\log(n))$

Programa

Foi criado um programa que calcula o tempo de inserção e busca de cada collection baseada em um arquivo txt que é inserido pelo usuário.

The screenshot shows a Java application window titled "Java Collections". It displays a table of insertion times for nine different Java collections. The collections are arranged in a 3x3 grid. Below the table, there are controls for file insertion and searching, including buttons for "Arquivo", "Buscar", and "Histograma".

Collection	Tempo de Inserção
ArrayList	0.002
LinkedList	0.003
HashSet	0.002
LinkedHashSet	0.006
TreeSet	0.01
HashMap	0.002
LinkedHashMap	0.003
TreeMap	0.007
B-Tree	0.016

Controls at the bottom:

- Inserir Arquivo de Texto :
- Histograma de Inserção :
- Buscar palavra :
-
- Histograma de Busca :

O tempo de inserção é atualizado automaticamente após a inserção de um arquivo, seu histograma também é liberado e será mostrado em tela se o usuário clicar no botão.

Para gerar o histograma de busca é preciso antes inserir uma palavra na caixa e em seguida iniciar uma busca, será mostrado em tela o valor da Key se esta se encontrar nas collections ou então será mostrado que a Key não foi encontrada.